WHAT IS CLAIMED IS:

1. An apparatus comprising:

a pointer storage configured to store a pointer to markup language data; and

a circuit coupled to the pointer storage, wherein the circuit is configured to parse the markup language data into one or more tokens, each token comprising one or more characters from the markup language data, wherein the circuit is configured to parse the markup language data responsive to one or more delimiters in the markup language data.

2. The apparatus as recited in claim 1 wherein the one or more delimiters are whitespace.

3. The apparatus as recited in claim 1 wherein the circuit is configured to generate a type for each of the tokens, wherein the type is dependent upon at least one of the delimiters delimiting the token.

4. The apparatus as recited in claim 3 wherein the type is an element name if the beginning delimiter is a less than character followed by a second character which is not an exclamation point character, a question mark character, or a forward slash character.

5. The apparatus as recited in claim 4 wherein the type of a first token which is a next token after an element name is an attribute name if the beginning delimiter of the first token is whitespace.

6. The apparatus as recited in claim 5 wherein the type of a second token which is the next token after an attribute name is an attribute value if the beginning delimiter of the second token is an equal sign character.

7. The apparatus as recited in claim 3 wherein the type is an instruction if the beginning delimiter is a less than character followed by a question mark character.

8. The apparatus as recited in claim 3 wherein the type is an end element if the beginning delimiter is a less than character followed by a forward slash character.

9. The apparatus as recited in claim 3 wherein the type is a comment if the beginning delimiter is a less than character followed by an exclamation point character followed by two dash characters.

10. The apparatus as recited in claim 3 wherein the type is an entity if the beginning delimiter is an ampersand character.

11. The apparatus as recited in claim 3 wherein the type is a declaration if the beginning delimiter is a less than character followed by an exclamation point character followed by one or more characters which are not two dashes immediately following the exclamation point character.

12. The apparatus as recited in claim 3 wherein the type is an abnormality if the token is not recognized by the circuit.

13. The apparatus as recited in claim 3 further comprising a table coupled to the circuit, the table comprising a plurality of entries, wherein each of the plurality of entries is configured to store a pointer to a software routine and corresponds to one of the types generated by the circuit.

14. The apparatus as recited in claim 13 further comprising an interface circuit coupled to receive software-generated commands, wherein the interface circuit is coupled to the

circuit, and wherein, in response to a command requesting a pointer corresponding to a token, the interface circuit is configured to return the pointer from the entry of the table which corresponds to the type of the token.

15. The apparatus as recited in claim 1 wherein the circuit is configured to detect an invalid character within the markup language data and is configured to signal an abnormality in response to the detecting.

16. The apparatus as recited in claim 1 wherein one of the delimiters is an end of file indication.

17. The apparatus as recited in claim 1 further comprising an interface circuit coupled to receive software-generated commands, wherein the interface circuit is coupled to the circuit.

18. The apparatus as recited in claim 17 wherein, in response to a first command which supplies a pointer to markup language data, the interface circuit is configured to cause the pointer storage to update with the pointer supplied by the first command.

19. The apparatus as recited in claim 17 wherein, in response to a second command, the circuit is configured to parse the next token in the markup language data.

20. The apparatus as recited in claim 19 wherein, in response to a third command, the circuit is configured to supply a pointer to the next token in the markup language data.

21. The apparatus as recited in claim 17 wherein the circuit is configured to update the pointer after the token has been delivered, to point to the next character in the markup language data after the end delimiter of the token.

22. The apparatus as recited in claim 1 further comprising a table coupled to the circuit, the table comprising a plurality of entries, wherein each of the plurality of entries is configured to store a string of one or more characters comprising a keyword, and wherein, the circuit is configured to detect whether or not a first token parsed from the markup language data matches one of the keywords in the table.

23. The apparatus as recited in claim 22 further comprising an interface circuit coupled to receive software-generated commands, wherein the interface circuit is coupled to the circuit, and wherein, in response to a command corresponding to the first token, the interface circuit is configured to return an indication of the entry storing the keyword which matches the first token.

24. A carrier medium carrying one or more data structures representing an apparatus, the apparatus comprising:

a pointer storage configured to store a pointer to markup language data; and

a circuit coupled to the pointer storage, wherein the circuit is configured to parse the markup language data into one or more tokens, each token comprising one or more characters from the markup language data, wherein the circuit is configured to parse the markup language data responsive to one or more delimiters in the markup language data.

25. The carrier medium as recited in claim 24 wherein the one or more delimiters are whitespace.

26. The carrier medium as recited in claim 24 wherein the circuit is configured to generate a type for each of the tokens, wherein the type is dependent upon at least one of the delimiters delimiting the token.

27. The carrier medium as recited in claim 26 wherein the type is an element name if the beginning delimiter is a less than character followed by a second character which is not an exclamation point character, a question mark character, or a forward slash character.

28. The carrier medium as recited in claim 27 wherein the type of a first token which is a next token after an element name is an attribute name if the beginning delimiter of the first token is whitespace.

29. The carrier medium as recited in claim 28 wherein the type of a second token which is the next token after an attribute name is an attribute value if the beginning delimiter of the second token is an equal sign character.

30. The carrier medium as recited in claim 26 wherein the type is an instruction if the beginning delimiter is a less than character followed by a question mark character.

31. The carrier medium as recited in claim 26 wherein the type is an end element if the beginning delimiter is a less than character followed by a forward slash character.

32. The carrier medium as recited in claim 26 wherein the type is a comment if the beginning delimiter is a less than character followed by an exclamation point character followed by two dash characters.

33. The carrier medium as recited in claim 26 wherein the type is an entity if the beginning delimiter is an ampersand character.

34. The carrier medium as recited in claim 26 wherein the type is a declaration if the beginning delimiter is a less than character followed by an exclamation point character followed by one or more characters which are not two dashes immediately following the

exclamation point character.

35. The carrier medium as recited in claim 26 wherein the type is an abnormality if the token is not recognized by the circuit.

36. The carrier medium as recited in claim 26 further comprising a table coupled to the circuit, the table comprising a plurality of entries, wherein each of the plurality of entries is configured to store a pointer to a software routine and corresponds to one of the types generated by the circuit.

37. The carrier medium as recited in claim 36 further comprising an interface circuit coupled to receive software-generated commands, wherein the interface circuit is coupled to the circuit, and wherein, in response to a command requesting a pointer corresponding to a token, the interface circuit is configured to return the pointer from the entry of the table which corresponds to the type of the token.

38. The carrier medium as recited in claim 24 wherein the circuit is configured to detect an invalid character within the markup language data and is configured to signal an abnormality in response to the detecting.

39. The carrier medium as recited in claim 24 wherein one of the delimiters is an end of file indication.

40. The carrier medium as recited in claim 24 further comprising an interface circuit coupled to receive software-generated commands, wherein the interface circuit is coupled to the circuit.

41. The carrier medium as recited in claim 40 wherein, in response to a first command which supplies a pointer to markup language data, the interface circuit is configured to

cause the pointer storage to update with the pointer supplied by the first command.

42. The carrier medium as recited in claim 40 wherein, in response to a second command, the circuit is configured to parse the next token in the markup language data.

43. The carrier medium as recited in claim 40 wherein, in response to a third command, the circuit is configured to supply a pointer to the next token in the markup language data.

44. The carrier medium as recited in claim 40 wherein the circuit is configured to update the pointer after the token has been delivered, to point to the next character in the markup language data after the end delimiter of the token.

45. The carrier medium as recited in claim 24 further comprising a table coupled to the circuit, the table comprising a plurality of entries, wherein each of the plurality of entries is configured to store a string of one or more characters comprising a keyword, and wherein, the circuit is configured to detect whether or not a first token parsed from the markup language data matches one of the keywords in the table.

46. The carrier medium as recited in claim 45 further comprising an interface circuit coupled to receive software-generated commands, wherein the interface circuit is coupled to the circuit, and wherein, in response to a command corresponding to the first token, the interface circuit is configured to return an indication of the entry storing the keyword which matches the first token.